AN13567

Optimizing Flashloader Download Speed on RT Platform with JLink

Rev. 0 — 18 February 2022

Application Note

Contents

1 Introduction

When using RT series chips in the debugging stage or the mass production stage, if the image downloaded to the external NOR FLASH is large, the downloading process is time-consuming. It affects the debugging efficiency or mass production efficiency. This application note introduces how to optimize the Flashloader algorithm to improve the download speed when using JLink to download image.

2 Flashloader algorithm introduction

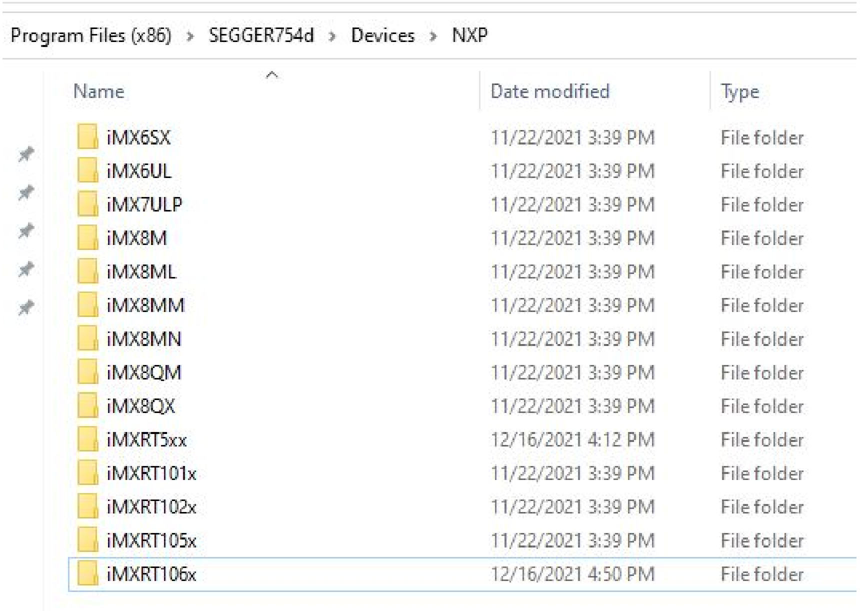
If using JLink to download the image on the RT platform, almost all Flashloader are currently developed based on the Open Flashloader model provided by SEGGER. For details about the framework and development process of SEGGER Flashloader, see Segger.

2.1 SEGGER supporting users to add target devices by themselves

SEGGER allows users to add support for new devices by themselves. There is no need to rely on newer SEGGER version and J-Link software. By default, the J-Link DLL comes with a built-in device database. The database defines which device names are known. For known devices, SEGGER provides a default burning and downloading algorithm. For unknown devices, add additional devices via an XML file called *JLinkDevices.xml*. Open the installation directory of *Segger...\Device\NXP\*, add the Flashloader algorithm file to the specified path. The file is in the XML format with the script syntax specified by SEGGER. If a known device is added to the XML file, JLink calls the Flashloader algorithm file in the XML file first.

```
<Database>
  <Device>
    <ChipInfo Vendor="..." Name="..." WorkRAMAddr="..." WorkRAMSize="..." Core="..." />
    <FlashBankInfo Name="..." BaseAddr="..." MaxSize="..." Loader="..." LoaderType="..."
AlwaysPresent="..." />
  </Device>
</Database>
```

To demonstrate how to improve the burning and downloading speed of the Flashloader, this article takes i.MX RT500 platform, i.MX RT1060 platform, and SEGGER 7.54d version as example. An iMXRT5xx folder and an iMXRT106x folder are added in the NXP folder, as shown in Figure 1. The folders contain *MIMXRT5XX_FLEXSPI.elf* file and *MIMXRT106X_FLEXSPI.elf* file. These two files are the optimized Flashloader executable files. They can be *.elf* files or *.FLM* files.



Program Files (x86) › SEGGER754d › Devices › NXP

Name	Date modified	Type
iMX6SX	11/22/2021 3:39 PM	File folder
iMX6UL	11/22/2021 3:39 PM	File folder
iMX7ULP	11/22/2021 3:39 PM	File folder
iMX8M	11/22/2021 3:39 PM	File folder
iMX8ML	11/22/2021 3:39 PM	File folder
iMX8MM	11/22/2021 3:39 PM	File folder
iMX8MN	11/22/2021 3:39 PM	File folder
iMX8QM	11/22/2021 3:39 PM	File folder
iMX8QX	11/22/2021 3:39 PM	File folder
iMXRT5xx	12/16/2021 4:12 PM	File folder
iMXRT101x	11/22/2021 3:39 PM	File folder
iMXRT102x	11/22/2021 3:39 PM	File folder
iMXRT105x	11/22/2021 3:39 PM	File folder
iMXRT106x	12/16/2021 4:50 PM	File folder

Figure 1. Flashloader algorithm file directory

2.2 Flashloader algorithm file generation platform

There are two ways to make Flashloader algorithm file of SEGGER:

- Use Keil uVision IDE. It has three disadvantages:
  - Requires a license.
  - No trial version.
  - Only supports Cortex-M core devices.
- Use SEGGER Embedded Studio IDE. It has two advantages:
  - A valid license is required for any commercial use of SEGGER Embedded Studio, but Open Flashloader is an exception. To debug and create Flashloader, an evaluation license can be used and no valid license is required.
  - Supports various cores, such as, Cortex-M, Cortex-A/R, and RISC-V.

2.3 Flashloader algorithm file generation method

SEGGER Embedded Studio provides the template of Open Flashloader. SEGGER Embedded Studio can generate the final Flashloader executable file. The Flashloader model consists of several important API functions, as shown in Table 1. This part of the API requires the user to implement the flash interface code of the corresponding MCU device. On the i.MX RT platform, it is based on the interaction code between FLEXSPI peripheral and NOR FLASH.

Table 1. Opening Flashloader API

Function name	Function
`Init`	To handle the initialization of the Flash module.
`UnInit`	To handle the deinitialization of the Flash module.
`Erase`	To erase one Flash sector. The `EraseSector` function can erase a single sector.

Table continues on the next page...

Table 1. Opening Flashloader API (continued)

Function name	Function
`Program`	To program one Flash page. The `ProgramPage` function can program a single page.
`EraseChip`	To erase the entire chip (Flash bank).
`Verify`	To compare a specified byte number of a provided data buffer with the content of the device.
`BlankCheck`	To check whether a memory region is blank.

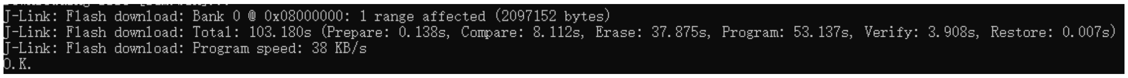
3 Factors affecting the download speed of Flashloader

This chapter introduces the factors that affect the download speed of Flashloader and ideas for optimizing the Open Flashloader model by using SEGGER Embedded Studio. All the experimental data in this chapter is based on RT500 EVK board and external NOR FLASH (GD25LE64C).

1. Download time components.

   a. Use the Jlink downloader to download a 2 M byte image to the external Flash to the RT500 EVK board through the **Commander** window.

   b. After the download, the log appears in the **Commander** window, as shown in Figure 2.



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 103.180s (Prepare: 0.138s, Compare: 8.112s, Erase: 37.875s, Program: 53.137s, Verify: 3.908s, Restore: 0.007s)
J-Link: Flash download: Program speed: 38 KB/s
O.K.
```

Figure 2. Download log in Commander window

Factors that affect the download speed are as shown in Table 2.

- **Erase** and **Program** take the largest proportion of time.
- **Compare** and **Verify** take a small proportion.
- **Prepare** and **Restore** take almost negligible time.

Therefore, to optimize the **Erase** and **Program** speed, optimize the download speed.

Table 2. Download time components

Image size	Prepare	Compare	Erase	Program	Verify	Restore	Total
2 M bytes	0.138 s	8.112 s	37.875 s	53.137 s	3.908 s	0.007 s	103.180 s

2. Factors related to download speed.

   Taking 4-wire QSPI NOR FLASH as an example, the operations of Flash include **Read**, **Write**, and **Erase**. Table 3 lists the CMD details of NOR FLASH.

Table 3. Flash CMD

Command name	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	n-Bytes
Write Enable	06H						
Write Disable	04H						
Volatile SR Write Enable	50H						

Table continues on the next page...

Table 3. Flash CMD (continued)

Command name	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	n-Bytes
Read Status Register	05H	(S7-S0)					(continuous)
Read Status Register-1	35H	(S15-S8)					(continuous)
Write Status Register	01H	S7-S0	S15-S8				
Read Data	03H	A23-A16	A15-A8	A7-A0	(D7-D0)	(Next byte)	(continuous)
Fast Read	0BH	A23-A16	A15-A8	A7-A0	dummy	(D7-D0)	(continuous)
Dual Output Fast Read	3BH	A23-A16	A15-A8	A7-A0	dummy	(D7-D0)	(continuous)
Dual I/O Fast Read	BBH	A23-A8	A7-A0 M7-M0	(D7-D0)			(continuous)
Quad Output Fast Read	6BH	A23-A16	A15-A8	A7-A0	dummy	(D7-D0)	(continuous)
Quad I/O Fast Read	EBH	A23-A0 M7-M0	dummy	(D7-D0)			(continuous)
Quad I/O Word Fast Read	E7H	A23-A0 M7-M0	dummy	(D7-D0)			(continuous)
Page Program	02H	A23-A16	A15-A8	A7-A0	D7-D0	Next byte	
Quad Page Program	32H	A23-A16	A15-A8	A7-A0	D7-D0		
Sector Erase	20H	A23-A16	A15-A8	A7-A0			
Block Erase (32 K)	52H	A23-A16	A15-A8	A7-A0			
Block Erase (64 K)	D8H	A23-A16	A15-A8	A7-A0			
Chip Erase	C7/60H						
Enable QPI	38H						
Enable Reset	66H						
Reset	99H						
Set Burst with Wrap	77H	W6-W4					
Program/Erase Suspend	75H						
Program/Erase Resume	7AH						

Read mode affects the efficiency of **Compare** and **Verify** for Flashloader. The CMD of Read includes Read Data, Fast Read, Quad Output Fast Read, Quad I/O Fast Read, Quad I/O Word Fast Read, as shown in Table 4.

The difference is in the process of communication between MCU and Flash:

- how many lines are used to send CMD commands;
- how many lines are used to send address information;
- how many lines are used for data interaction.

If it is a 4-line QSPI NOR FLASH, use the **1_4_4** mode.

Table 4. Read CMD modes

CMD	Read mode
Read Data(0x03)/Fast Read	1_1_1
Quad Output Fast Read	1_1_4
Quad I/O Fast Read	1_4_4

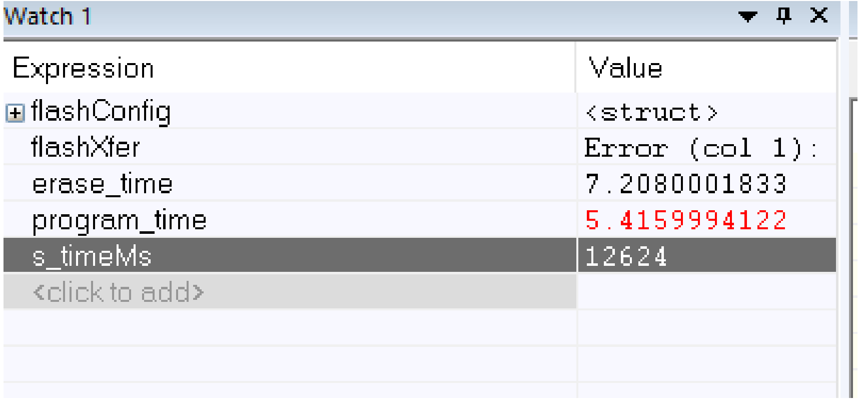
Except for Chip Erase, Erase CMD contains three types: Sector Erase, Block Erase (32 K), Block Erase (64 K). Erase mode can work in SPI mode or QPI mode. It is enough to work in SPI mode. To erase a large area, Block Erase is more efficient than Sector Erase.

Program CMD contains two types: Page Program and Quad Page Program. It is enough to work in Page Program.

In addition to Read mode, Erase mode, and Program mode, the other important factor that affects the download efficiency of Flashloader is the clock frequency of Flash. The impact is limited. To prove this point, see the following experiment.

a. On the RT500 EVK board, replace the external NOR FLASH with GD25LE64C, debug a pre-prepared Flashloader algorithm under the IAR, and use the MCU system clock to count the erasing time. The erasing area is 2 MB of external NOR FLASH space. To ensure the comparability of the experimental results, keep same starting address of the Flash and use the same `.bin` file for the 2 MB image. Use **Block Erase (64 K)** for Erase CMD, **Page Program** for Program CMD, and change only the CLK of FlexSPI. The results are as shown in Figure 3 and Figure 4.

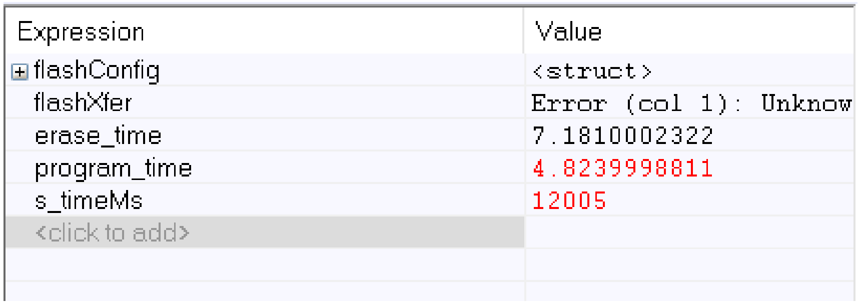
   - Set the CLK frequency of FlexSPI to 30 M, and the erasing time is 12.624 s.



Expression	Value
⊞ flashConfig	<struct>
flashXfer	Error (col 1):
erase_time	7.2080001833
program_time	5.4159994122
s_timeMs	12624
<click to add>	

Figure 3. Erase time and Program time

   - Set the CLK frequency of FlexSPI to 100 M, and the erasing time is 12.005 s.

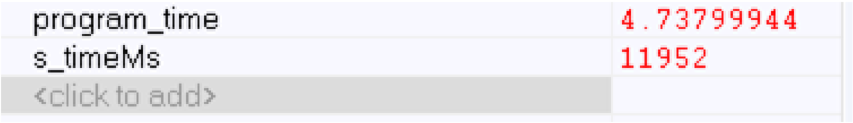


Expression	Value
⊞ flashConfig	<struct>
flashXfer	Error (col 1): Unknow
erase_time	7.1810002322
program_time	4.8239998811
s_timeMs	12005
<click to add>	

Figure 4. Erase time and Program time

The first conclusion is that the frequency of Flash has little effect on the erase speed and program speed.

b. Set the CLK frequency of FlexSPI to 100 M, change the Program CMD mode from **Page Program** to **Quad Page Program**, and keep other conditions unchanged. The result is that the erasing time is 11.952 s, as shown in Figure 5.



program_time	4.73799944
s_timeMs	11952
<click to add>	

Figure 5. Erase time and Program time

The second conclusion is: Program mode has little effect on the speed of the entire process of writing a page.

c. Table 5 lists the typical values of erase time and program time. For details, see the data sheet of Flash.

Table 5. Maximum and typical value of erase time and program time

t_{PP}	Page Programming Time		0.7	5	ms
t_{SE}	Sector Erase Time		90	600	ms
t_{BE1}	Block Erase Time (32 K Bytes)		0.3	1.5	s
t_{BE2}	Block Erase Time (64 K Bytes)		0.45	3.0	s
t_{CE}	Chip Erase Time (GD25LE64C)		30	90	s

Comparing the theoretical value, IAR debug test data, and JLink Commander test data, the debug of the Flashloader algorithm in IAR is very close to or even smaller than the typical value of FLASH data sheet. However, when the Flashloader algorithm is called by Jlink Commander, the erase and program efficiency is very low, as shown in Table 6.

Table 6. Comparison of erase time and program time

Tool chain	Image size (MB)	Erase (s)	Program (s)
SEGGER commander	2	37.875	53.137
Test data on IAR	2	7.208	5.416
Typical value/Maximum value	2	14.4 (typical)/96.0 (max)	5.734 (typical)/40.960 (max)

The third conclusion is that the biggest bottleneck affecting the flashloader download speed is the scheduling of the Flashloader algorithm by Jlink Commander or the interaction process between Jlink Commander and the Flashloader program running on the MCU side.

3. Open Flashloader Model Framework.

The download rate is related to the scheduling process of the Flashloader running on the MCU side by Jlink through the SWD/JTAG interface, mainly in the parts of **Compare**, **Erase**, **Program**, and **Verify**.

As mentioned above, Keil or SEGGER Embedded Studio IDE can generate the Flashloader executable file. The Erase and Program API interfaces in Flashloader model of Keil have limitations on download efficiency, so it is recommended to use SEGGER Embedded Studio.

The Open Flashloader model provided by SEGGER Embedded Studio provides SEGGER_OPEN_Program API interface and Program API interface. Compared with the traditional Program API interface, the SEGGER_OPEN_Program API interface can achieve higher throughput, as shown in Figure 6.

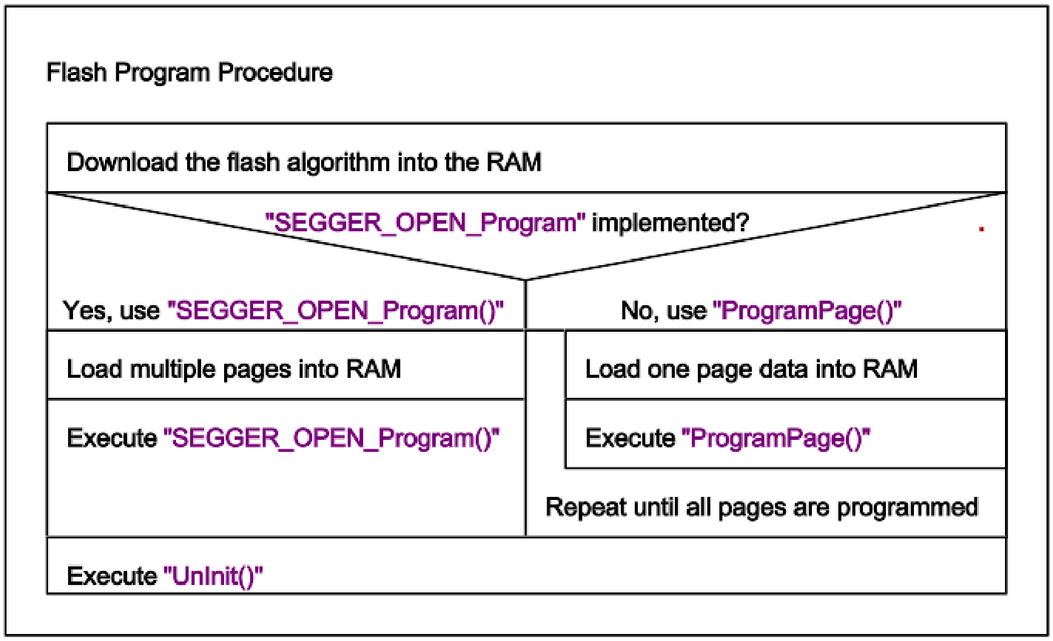

Flash Program Procedure
Download the flash algorithm into the RAM
"SEGGER_OPEN_Program" implemented?
Yes, use "SEGGER_OPEN_Program()"
No, use "ProgramPage()"
Load multiple pages into RAM
Execute "SEGGER_OPEN_Program()"
Load one page data into RAM
Execute "ProgramPage()"
Repeat until all pages are programmed
Execute "UnInit()"

Figure 6. Open Flashloader program procedure

Similarly, the Open Flashloader provided by SEGGER Embedded Studio provides SEGGER_OPEN_Erase API interface and Erase API interface. The Erase interface is used to erase one or more sector areas. Considering that different types of Flash have different sector sizes, the sector size is user-configurable.

Through the above analysis, to erase a large area, using Block erase is more efficient than Sector erase. The Erase interface API interface is user-defined, so users can use the sector size provided in the Open Flashloader as the block size. To improve the efficiency of erase, use block erasing instead of sector erasing in the underlying code.

The Open Flashloader model provides API interfaces, such as **Verify**, **Read**, and **Erase_Chip**. To optimize the download algorithm, enable the TURBO mode. These methods are the entry points for optimizing the download speed.

4 Flashloader download speed optimization

This chapter verifies some optimization strategies to improve download speed based on RT500 EVK board and RT1060 EVKB board. To compare the optimization results of the Flashloader algorithm, replace the Flash on the RT500 EVK board and RT1060 EVKB board with IS25WP064. Table 7 lists the theoretical values of Erase and Program in the IS25WP064 data sheet. According to the typical value, the typical duration of Sector Erase, Block Erase (64 K), and Program are respectively 35.84 s, 4.8 s, and 1.638 s for the 2 MB Flash space. The following sections compare the experimental test data with the theoretical data. In the actual downloading process of Flash using JLink, there must be other operation durations.

Table 7. IS25WP064 erase time and program time typical value

Symbol	Parameter		Min.	Type	Max.	Unit
t_{ec}	Sector Erase Time (4 kB)			70	300	ms
	Block Erase Time (32 kB)			0.1	0.5	s
	Block Erase Time (64 kB)			0.15	1.0	s
	Chip Erase Time	32 MB		8	23	s
		64 MB		16	45	

Table continues on the next page...

Table 7. IS25WP064 erase time and program time typical value (continued)

Symbol	Parameter		Min.	Type	Max.	Unit
		128 MB		30	90	
t_{pp}	Page Program Time			0.2	0.8	ms

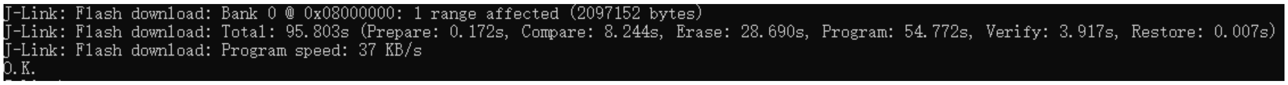
4.1 Flashloader optimization based on RT500

SEGGER Embedded Studio or Keil generate the Flashloader executable file. Erase API interface uses the **Erase** interface and Program API interface uses the **SEGGER_OPEN_Program** interface. The sector size of IS25WP064 is 4 kB, and the page size is 256 bytes. SEGGER_OPEN_Program API interface can load multiple pages into RAM for caching at one time. This value is configurable. The configurations in the test are as below.

- Define the size of a page in the SEGGER_OPEN_Program API interface as the variable `multi_page_size`.
- Define the size of a page in the Program interface as the variable `single_page_size`.
- Define the size of each erase in the interface as the variable `single_erase_size`.
- Set the CLK of FlexSPI to 30 M.
- The flash space size for the operation is `[0x0800 0000, 0x0820 0000]`, 2 MB.

Tests are described as below.

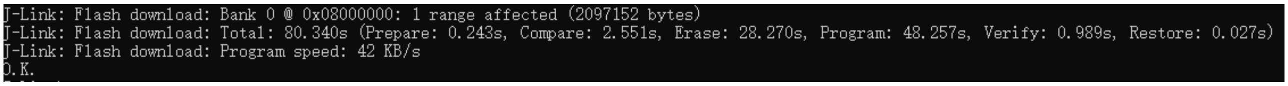
- Test 1
  - Generation Platform: Keil
  - Erase Interface: Erase
  - Program Interface: Program
  - single_page_size: 256 Byte
  - single_erase_size: 4 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 95.803s (Prepare: 0.172s, Compare: 8.244s, Erase: 28.690s, Program: 54.772s, Verify: 3.917s, Restore: 0.007s)
J-Link: Flash download: Program speed: 37 KB/s
O.K.
```

Figure 7. Test 1

- Test 2
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: Program
  - `single_page_size`: 256 Byte
  - `single_erase_size`: 4 kB
  - TURBO MODE: Disable

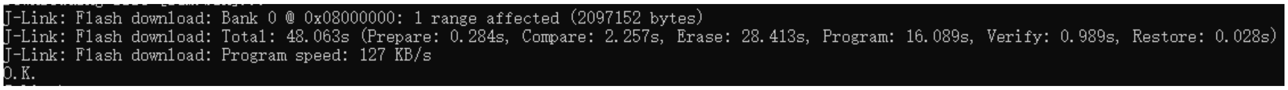


```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 80.340s (Prepare: 0.243s, Compare: 2.551s, Erase: 28.270s, Program: 48.257s, Verify: 0.989s, Restore: 0.027s)
J-Link: Flash download: Program speed: 42 KB/s
O.K.
```

Figure 8. Test 2

- Test 3

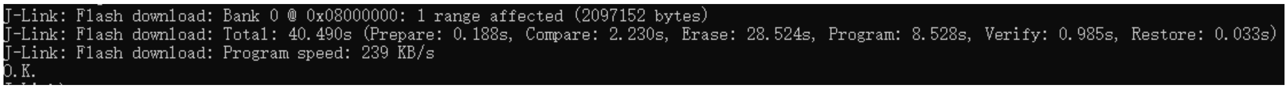
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: Program
  - `single_page_size`: 1 kB
  - `single_erase_size`: 4 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 48.063s (Prepare: 0.284s, Compare: 2.257s, Erase: 28.413s, Program: 16.089s, Verify: 0.989s, Restore: 0.028s)
J-Link: Flash download: Program speed: 127 KB/s
O.K.
```

Figure 9. Test 3

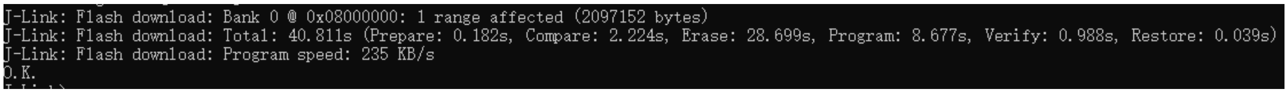
- Test 4
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: Program
  - `single_page_size`: 4 kB
  - `single_erase_size`: 4 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 40.490s (Prepare: 0.188s, Compare: 2.230s, Erase: 28.524s, Program: 8.528s, Verify: 0.985s, Restore: 0.033s)
J-Link: Flash download: Program speed: 239 KB/s
O.K.
```

Figure 10. Test 4

- Test 5
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: Program
  - `single_page_size`: 8 kB
  - `single_erase_size`: 4 kB
  - TURBO MODE: Disable

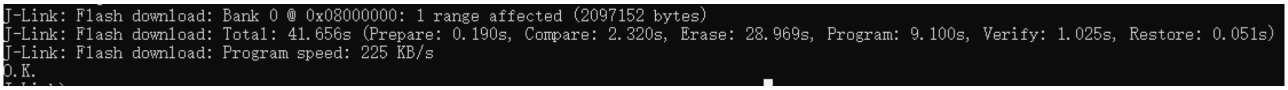


```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 40.811s (Prepare: 0.182s, Compare: 2.224s, Erase: 28.699s, Program: 8.677s, Verify: 0.988s, Restore: 0.039s)
J-Link: Flash download: Program speed: 235 KB/s
O.K.
```

Figure 11. Test 5

- Test 6
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: Program
  - `single_page_size`: 16 kB
  - `single_erase_size`: 4 kB

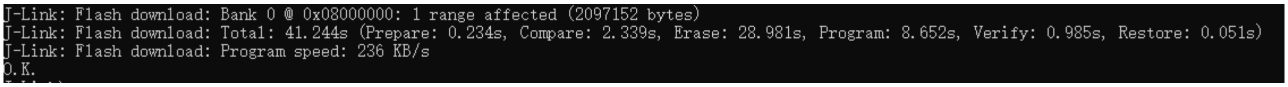
- TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 41.656s (Prepare: 0.190s, Compare: 2.320s, Erase: 28.969s, Program: 9.100s, Verify: 1.025s, Restore: 0.051s)
J-Link: Flash download: Program speed: 225 KB/s
O.K.
```

Figure 12. Test 6

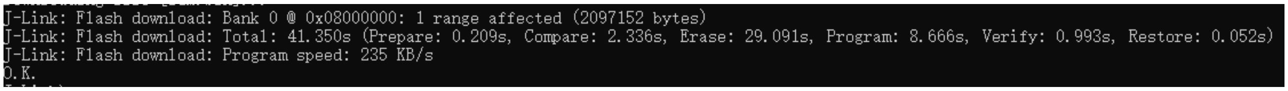
- Test 7
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - `multi_page_size`: 256 Byte
  - `single_erase_size`: 4 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 41.244s (Prepare: 0.234s, Compare: 2.339s, Erase: 28.981s, Program: 8.652s, Verify: 0.985s, Restore: 0.051s)
J-Link: Flash download: Program speed: 236 KB/s
O.K.
```

Figure 13. Test 7

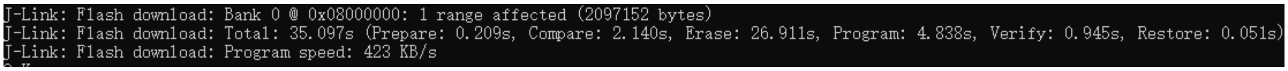
- Test 8
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - `multi_page_size`: 1 kB
  - `single_erase_size`: 4 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 41.350s (Prepare: 0.209s, Compare: 2.336s, Erase: 29.091s, Program: 8.666s, Verify: 0.993s, Restore: 0.052s)
J-Link: Flash download: Program speed: 235 KB/s
O.K.
```

Figure 14. Test 8

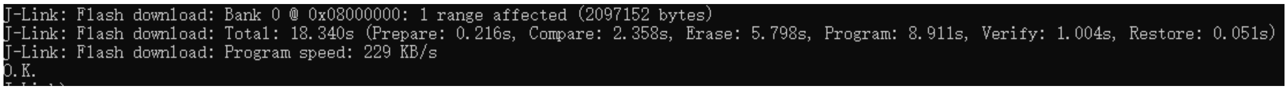
- Test 9
  - Generation Platform: SEGGER Embedded Studio (SES)
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - `multi_page_size`: 256 Byte
  - `single_erase_size`: 4 kB
  - TURBO MODE: Enable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 35.097s (Prepare: 0.209s, Compare: 2.140s, Erase: 26.911s, Program: 4.838s, Verify: 0.945s, Restore: 0.051s)
J-Link: Flash download: Program speed: 423 KB/s
```

Figure 15. Test 9

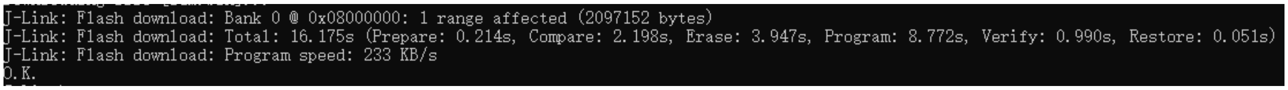
- Test 10
  - Generation Platform: SEGGER Embedded Studio（SES）
  - Erase Interface: Erase
  - Program interface: SEGGER_OPEN_Program
  - `multi_page_size`: 4 kB
  - `single_erase_size`: 32 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 18.340s (Prepare: 0.216s, Compare: 2.358s, Erase: 5.798s, Program: 8.911s, Verify: 1.004s, Restore: 0.051s)
J-Link: Flash download: Program speed: 229 KB/s
O.K.
```

Figure 16. Test 10

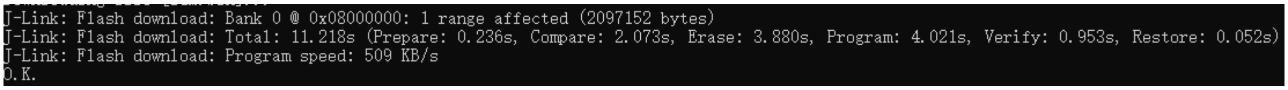
- Test 11
  - Generation Platform: SEGGER Embedded Studio（SES）
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - `multi_page_size`: 4 kB
  - `single_erase_size`: 64 kB
  - TURBO MODE: Disable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 16.175s (Prepare: 0.214s, Compare: 2.198s, Erase: 3.947s, Program: 8.772s, Verify: 0.990s, Restore: 0.051s)
J-Link: Flash download: Program speed: 233 KB/s
O.K.
```

Figure 17. Test 11

- Test 12
  - Generation Platform: SEGGER Embedded Studio（SES）
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - `multi_page_size`: 4 kB
  - `single_erase_size`: 64 kB
  - TURBO MODE: Enable

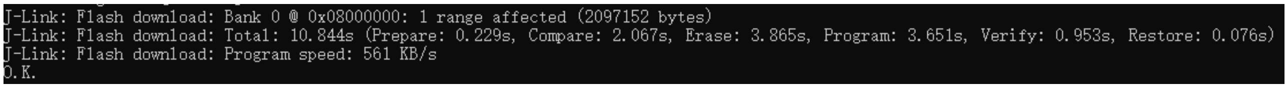


```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 11.218s (Prepare: 0.236s, Compare: 2.073s, Erase: 3.880s, Program: 4.021s, Verify: 0.953s, Restore: 0.052s)
J-Link: Flash download: Program speed: 509 KB/s
O.K.
```

Figure 18. Test 12

- Test 13
  - Generation Platform: SEGGER Embedded Studio（SES）
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - `multi_page_size`: 16 kB

- — `single_erase_size`: 64 kB
- — TURBO MODE: Enable



```
J-Link: Flash download: Bank 0 @ 0x08000000: 1 range affected (2097152 bytes)
J-Link: Flash download: Total: 10.844s (Prepare: 0.229s, Compare: 2.067s, Erase: 3.865s, Program: 3.651s, Verify: 0.953s, Restore: 0.076s)
J-Link: Flash download: Program speed: 561 KB/s
O.K.
```

Figure 19. Test 13

Table 8. Flashloader test data on RT500 EVK board

Test	Generate tool	Image size (MB)	Single erase size (kB)	Open program interface	Program size/ Page mode	Turbo mode	Prepare (s)	Compare (s)	Erase (s)	Program (s)	Verify (s)	Restore (s)	Total (s)
1	Keil	2	4	Disable	256 B single	Disable	0.172	8.244	28.690	54.772	3.917	0.007	95.803
2	SES	2	4	Disable	256 B single	Disable	0.243	2.551	28.270	48.257	0.989	0.027	80.340
3	SES	2	4	Disable	1 kB single	Disable	0.284	2.257	28.413	16.089	0.989	0.028	48.063
4	SES	2	4	Disable	4 kB single	Disable	0.188	2.230	28.524	8.528	0.985	0.033	40.490
5	SES	2	4	Disable	8 kB single	Disable	0.182	2.224	28.699	8.677	0.988	0.039	40.811
6	SES	2	4	Disable	16 kB single	Disable	0.190	2.320	28.969	9.100	1.025	0.051	41.656
7	SES	2	4	Enable	256 B multiple	Disable	0.234	2.339	28.981	8.652	0.985	0.051	41.244
8	SES	2	4	Enable	1 kB multiple	Disable	0.209	2.336	29.091	8.666	0.993	0.052	41.350
9	SES	2	4	Enable	256 B multiple	Enable	0.209	2.140	26.911	4.838	0.945	0.051	35.097
10	SES	2	32	Enable	4 kB multiple	Disable	0.216	2.358	5.798	8.911	1.004	0.051	18.340
11	SES	2	64	Enable	4 kB multiple	Disable	0.214	2.198	3.947	8.772	0.990	0.051	16.175

Table continues on the next page...

Table 8. Flashloader test data on RT500 EVK board (continued)

Test	Generate tool	Image size (MB)	Single erase size (kB)	Open program interface	Program size/ Page mode	Turbo mode	Prepare (s)	Compare (s)	Erase (s)	Program (s)	Verify (s)	Restore (s)	Total (s)
12	SES	2	64	Enable	4 kB multiple	Enable	0.236	2.073	3.880	4.021	0.953	0.052	11.218
13	SES	2	64	Enable	16 kB multiple	Enable	0.229	2.067	3.865	3.651	0.953	0.076	10.844

With the experimental data shown in Table 8, the following conclusions can be drawn:

1. The Flashloader generated by SEGGER Embedded Studio is more efficient to download than the Flashloader generated by Keil.
2. With the Program API interface, increasing the Page Size loaded into RAM each time has a limited improvement in download speed.
3. Flash is written in a Page in the Program or SEGGER_OPEN_Program interface of Flash. In the experiment, changing `single_page_size` or `multi_page_size` affects the size of the buffer each time JLink loads into the MCU memory, but finally the underlying Program operation must be based on one page. The improvement of the SEGGER_OPEN_Program interface over the Program interface is that the size of the buffer each time JLink loads into the MCU memory is several times the size of the original buffer each time it is loaded.
4. Enabling TURBO Mode can improve the download speed of Flashloader, so it is recommended to enable this function.
5. When using the SEGGER_OPEN_Program API interface and enabling TURBO Mode, the larger the `multi_page_size` is, the faster the program speed is. Therefore, set `multi_page_size` to 4 kB.
6. For Erase interface API, this article does not recommend using the SEGGER_OPEN_Erase interface. To improve the speed of Erase, replace the Sector Erase solution with the Block Erase solution. This replacement has limitations.
   - The Flash size erased by the user is large enough (preferably several times the Block Size).
   - The starting address of the downloaded Flash is aligned with the block size.

   It does not matter whether these two conditions are met or not. The solution is to set the `single_erase_size` size to the block size. In the underlying erase driver code, the erased area is divided into a combination of multiple blocks and multiple sectors through an algorithm. It then calls the `Block_Erase CMD` and `Sector_Erase CMD` underlying drivers respectively to operate.

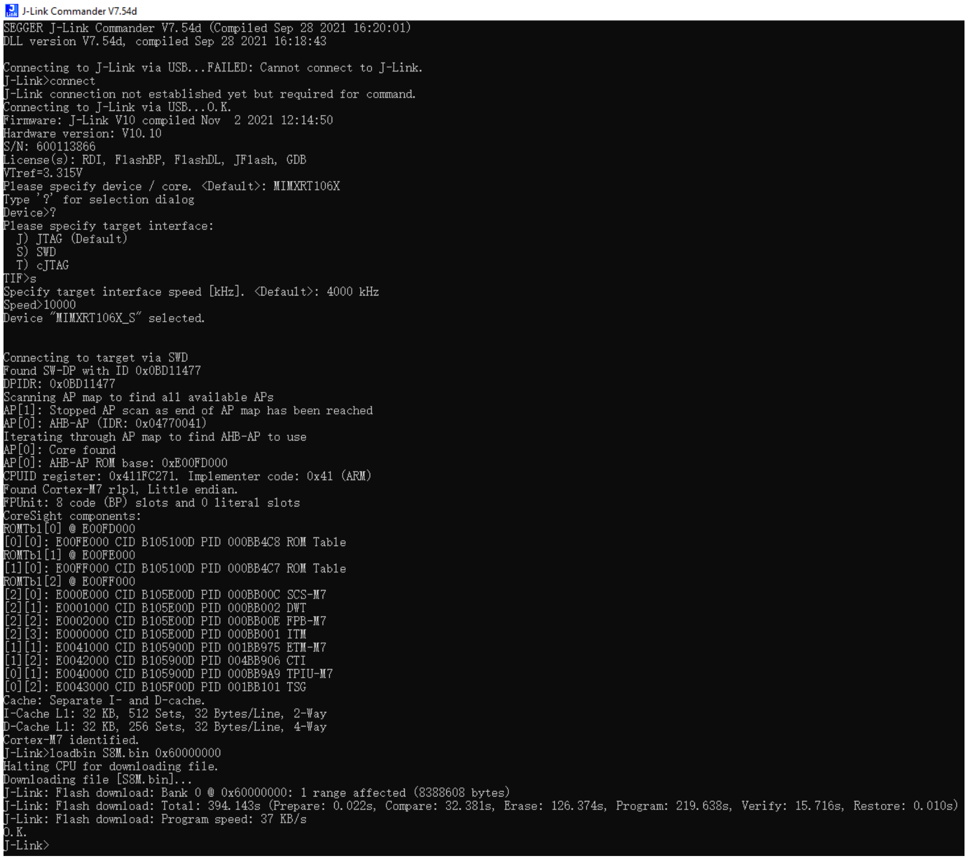
4.2 Flashloader optimization based on RT1060

This section uses the RT1060 EVKB board and external NOR FLASH to test the Flashloader download speed under different strategies. The default Flash model on the RT1060 EVKB board is IS25WP064. With the conclusion in Flashloader optimization based on RT500, this section skips the experiment in Flashloader optimization based on RT500 and directly test. Set the CLK of FlexSPI to 30 M. The size of the flash space for the operation is [0x6000 0000, 0x6080 0000], 8 MB.

Tests are described as below.

- Test 1
  - — Generation Platform: Keil
  - — Erase Interface: Erase
  - — Program Interface: Program

- single_page_size: 256 Byte
- single_erase_size: 4 KB
- TURBO MODE: Disable



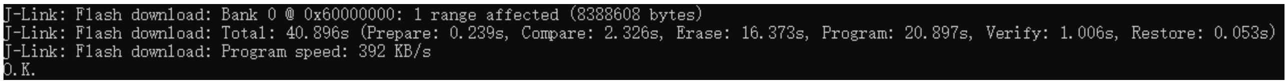
```
SEGGER J-Link Commander V7.54d (Compiled Sep 28 2021 16:20:01)
DLL version V7.54d, compiled Sep 28 2021 16:18:43

Connecting to J-Link via USB...FAILED: Cannot connect to J-Link.
J-Link>connect
J-Link connection not established yet but required for command.
Connecting to J-Link via USB...O.K.
Firmware: J-Link V10 compiled Nov  2 2021 12:14:50
Hardware version: V10.10
S/N: 600113866
License(s): RDI, FlashBP, FlashDL, JFlash, GDB
VTref=3.315V
Please specify device / core. <Default>: MIMXRT106X
Type '?' for selection dialog
Device>?
Please specify target interface:
  J) JTAG (Default)
  S) SWD
  T) cJTAG
TIF>s
Specify target interface speed [kHz]. <Default>: 4000 kHz
Speed>10000
Device "MIMXRT106X_S" selected.


Connecting to target via SWD
Found SW-DP with ID 0x0BD11477
DPIDR: 0x0BD11477
Scanning AP map to find all available APs
AP[1]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x04770041)
Iterating through AP map to find AHB-AP to use
AP[0]: Core found
AP[0]: AHB-AP ROM base: 0xE00FD000
CPUID register: 0x411FC271. Implementer code: 0x41 (ARM)
Found Cortex-M7 r1p1, Little endian.
FPUnit: 8 code (BP) slots and 0 literal slots
CoreSight components:
ROMTbl[0] @ E00FD000
[0][0]: E00FE000 CID B105100D PID 000BB4C8 ROM Table
ROMTbl[1] @ E00FE000
[1][0]: E00FF000 CID B105100D PID 000BB4C7 ROM Table
ROMTbl[2] @ E00FF000
[2][0]: E000E000 CID B105E00D PID 000BB00C SCS-M7
[2][1]: E0001000 CID B105E00D PID 000BB002 DWT
[2][2]: E0002000 CID B105E00D PID 000BB00E FPB-M7
[2][3]: E0000000 CID B105E00D PID 000BB001 ITM
[1][1]: E0041000 CID B105900D PID 001BB975 ETM-M7
[1][2]: E0042000 CID B105900D PID 004BB906 CTI
[0][1]: E0040000 CID B105900D PID 000BB9A9 TPIU-M7
[0][2]: E0043000 CID B105F00D PID 001BB101 TSG
Cache: Separate I- and D-cache.
I-Cache L1: 32 KB, 512 Sets, 32 Bytes/Line, 2-Way
D-Cache L1: 32 KB, 256 Sets, 32 Bytes/Line, 4-Way
Cortex-M7 identified.
J-Link>loadbin S8M.bin 0x60000000
Halting CPU for downloading file.
Downloading file [S8M.bin]...
J-Link: Flash download: Bank 0 @ 0x60000000: 1 range affected (8388608 bytes)
J-Link: Flash download: Total: 394.143s (Prepare: 0.022s, Compare: 32.381s, Erase: 126.374s, Program: 219.638s, Verify: 15.716s, Restore: 0.010s)
J-Link: Flash download: Program speed: 37 KB/s
O.K.
J-Link>
```

Figure 20. Test 1

- Test 2
  - Generation Platform: SEGGER Embedded Studio（SES）
  - Erase Interface: Erase
  - Program Interface: SEGGER_OPEN_Program
  - multi_page_size: 4 KB
  - single_erase_size: 64 KB
  - TURBO MODE: Enable



```
J-Link: Flash download: Bank 0 @ 0x60000000: 1 range affected (8388608 bytes)
J-Link: Flash download: Total: 40.896s (Prepare: 0.239s, Compare: 2.326s, Erase: 16.373s, Program: 20.897s, Verify: 1.006s, Restore: 0.053s)
J-Link: Flash download: Program speed: 392 KB/s
O.K.
```

Figure 21. Test 2

Table 9. Flashloader test data on RT1060 EVKB board

Test	Generate tool	Image size (MB)	Single erase size (kB)	Open program interface	Program size/ Page mode	Turbo mode	Prepare (s)	Compare (s)	Erase (s)	Program (s)	Verify (s)	Restore (s)	Total (s)
1	Keil	2	4	Disable	256 B single	Disable	0.022	32.381	126.374	219.638	15.716	0.010	394.143
2	SES	2	64	Enable	4 KB multiple	Enable	0.239	2.326	16.373	20.897	1.006	0.053	40.896

As shown in Table 9, when using the SEGGER_OPEN_Program API interface, the download speed is greatly improved, if TURBO mode is enabled, `multi_page_size` is set to 4 KB, and Block Erase solution is used to erase.

4.3 Flashloader download speed optimization results

Table 10 describes the download speed optimization results of Flashloader on RT500 EVK and RT1060 EVKB. The optimized download speed is 8.5 times and 9.6 times that of the original unoptimized. The experimental results prove that the optimization strategy is feasible.

Table 10. Download speed optimization data results of Flashloader on RT500 and RT1060

RT platform	Promotion	Test	Image size	Prepare	Compare	Erase	Program	Verify	Restore	Total
RT500	No optimization	1	2 MB	0.172 s	8.244 s	28.690 s	54.772 s	3.917 s	0.007 s	95.803 s
	Optimized	12	2 MB	0.236 s	2.073 s	3.880 s	4.021 s	0.953 s	0.052 s	11.218 s
	Promotion				398 %	739 %	1362 %	411 %		854 %
RT1060	No optimization	1	8 MB	0.022 s	32.381 s	126.374 s	219.638 s	15.716 s	0.010 s	394.143 s
	Optimized	2	8 MB	0.239 s	2.326 s	16.373 s	20.897 s	1.006 s	0.053 s	40.896 s
	Promotion				1392 %	772 %	1051 %	1562 %		964 %

As shown in Table 10, the efficiency of Compare and Verify on RT1060 is much higher than that of RT500. The main factor affecting them is the performance difference between MCU reading Flash data to internal RAM. RT1060 is a Cortex-M7 core and RT500 is Cortex-M33 core. RT1060 has 32 KB, L1 level I-cache and D-cache, and higher frequency (600 M). The features greatly improve the reading efficiency of Flash.

In addition, the Flashloader algorithm used in the test can automatically identify and support most NOR FLASH. It supports all i.MX RT platforms. This algorithm is used for RT-UFL (RT-UFL is a general Flash download algorithm project on RT, covering most Flash models). For details, see https://github.com/JayHeng/RT-UFL.

5 References

1. SEGGER Open Flashloader
2. MIMXRT500-EVK Schematic(Rev E1)

3. MIMXRT1060-EVKB Schematic(Rev B)
4. GD25LE64C Data Sheet
5. IS25WP064A Data Sheet
6. RT-UFL

6 Revision history

Rev.	Date	Description
0	18 February 2022	Initial release

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

arm

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 18 February 2022
Document identifier: AN13567